PegasusIO Documentation

Release 0.7.1

Bo Li, Yiming Yang

Aug 08, 2022

CONTENTS

PegasusIO is the IO package for Pegasus.

Read documentation

CHAPTER

ONE

VERSION 0.7.1 AUGUST 7, 2022

- Bug fix in `to_anndata` function. [PR #97]

CHAPTER

TWO

VERSION 0.7.0 JULY 25, 2022

- The default count matrix of a `UnimodalData` object now has key `counts` instead of `X`.
- Add `uid` option to `UnimodalData` constructor.
- Add `get_uid` function to `UnimodalData` class.
-

CHAPTER

THREE

VERSION 0.6.2 JULY 5, 2022

- In `read_input` function, add `transpose` option to transpose the loaded count matrix. Only works for CSV or TSV-format files.

CHAPTER

FOUR

VERSION 0.6.1 MAY 18, 2022

- Make the generated `h5` format count matrix readable by `read10xCount` function in R DropletUtil package. [PR #93]

CHAPTER

FIVE

VERSION 0.6.0 MAY 14, 2022

- `write_output` function supports 10x hdf5 format. [PR #92]

CHAPTER

SIX

VERSION 0.5.1 FEBRUARY 10, 2022

- Make PegasusIO work with Zarr v2.11.0.
- Bug fix in quality control. [PR #89 by hoondy]

CHAPTER

SEVEN

VERSION 0.5.0 JANUARY 24, 2022

- Add support on 10x Visium spatial data
  - Read the data folder by `read_input` function with `file_type="visium"` option.
  - Write 10x Visium data to Zarr format by `write_output` function with output file name of `.zarr.zip` extension.

CHAPTER

EIGHT

VERSION 0.4.1 NOVEMBER 4, 2021

- Fix issues on `UnimodalData` object construction.

CHAPTER

NINE

VERSION 0.4.0 OCTOBER 19, 2021

- Add `obsp` and `varp` fields to store graph representation in terms of square matrices.
- Allow copy from View of `AnnData`.
- In `MultimodalData`, add `register_attr` function to register an attribute of a specified type in `obs` or `obsm` fields. This can be useful for adding information like gene signatures, etc.

CHAPTER

TEN

VERSION 0.3.1 JULY 16, 2021

- For `aggregate_matrices` function, allow sample-specific filtration on minimum number of UMIs (`nUMI` column in sample sheet) and minimum number of genes (`nGene` column in sample sheet), which would overwrite the corresponding parameters of the function for these samples.

CHAPTER

ELEVEN

VERSION 0.3.0 JULY 6, 2021

- Add support for composite list (e.g. `[0, pd.DataFrame, np.ndarray]`) in `data.uns` field for Zarr read/write.

CHAPTER

TWELVE

VERSION 0.2.14 JUNE 28, 2021

- Add parameter `uns_white_list` in `filter_data` function to keep QC statistics if needed.

CHAPTER

THIRTEEN

VERSION 0.2.13 JUNE 24, 2021

- The `aggregate_matrices` function now accepts sample sheet in Python dictionary format besides a CSV file path string. See details in its description in API panel.

CHAPTER

FOURTEEN

VERSION 0.2.12 MAY 28, 2021

- Bug fix.

CHAPTER

FIFTEEN

VERSION 0.2.11 MAY 17, 2021

- Bug fix.

CHAPTER

SIXTEEN

VERSION 0.2.10 FEBRUARY 2, 2021

- Feature enhancement.

CHAPTER

SEVENTEEN

VERSION 0.2.9 DECEMBER 25, 2020

- Fix a bug for caching percent mito rate.
- Improve *write_mtx* function.

CHAPTER

EIGHTEEN

VERSION 0.2.8 DECEMBER 7, 2020

- Add support on loading `loom` file with Seurat-style cell barcode and feature key names.
- Bug fix: resolve an issue on count matrix dimension inconsistency with feature metadata on data aggregation, when last feature has `0` count across all cell barcodes. Thanks to Mikhail Alperovich for reporting this issue.
- Other bug fix and performance improvements.

CHAPTER

NINETEEN

VERSION 0.2.7 OCTOBER 13, 2020

- Add support for Nanostring GeoMx data format.
- Fix bugs.

CHAPTER

TWENTY

VERSION 0.2.6 SEPTEMBER 28, 2020

Fix bug in SCP compatible output generation.

CHAPTER

TWENTYONE

VERSION 0.2.5 AUGUST 19, 2020

Adjustment for Pegasus command usage.

CHAPTER

TWENTYTWO

VERSION 0.2.2 JUNE 16TH, 2020

Fix bugs in data aggregation.

CHAPTER

TWENTYTHREE

VERSION 0.2.1 JUNE 8TH, 2020

Fix bug in processing single `h5` file.

CHAPTER

TWENTYFOUR

VERSION 0.2.0 JUNE 7TH, 2020

Initial release.

24.1 Installation

PegasusIO is released on PyPI, and can be installed using pip:

```
pip install pegasusio
```

To install its development version, do the following:

```
git clone https://github.com/lilab-bcb/pegasusio.git
cd pegasusio
pip install -e .
```

24.2 API

Import *PegasusIO* to Python environment by:

```
import pegasusio as io
```

24.2.1 Read and Write

`infer_file_type`(input_file)	Infer file format from input_file name This function infer file type by inspecting the file name.
`read_input`(input_file[, file_type, mode, ...])	Load data into memory.
`write_output`(data, output_file[, file_type, ...])	Write data back to disk.
`aggregate_matrices`(csv_file[, restrictions, ...])	Aggregate channel-specific count matrices into one big count matrix.

24.3 Use PegasusIO as a command line tool

PegasusIO can be used as a command line tool. Type:

```
pegasus -h
```

to see the help information:

```
Usage:
  pegasus <command> [<args>...]
  pegasus -h | --help
  pegasus -v | --version
```

PegasusIO currently has only one sub-command:

- `aggregate_matrix`: Aggregate sample count matrices into a single count matrix. It also enables users to import metadata into the count matrix.

24.3.1 pegasusio aggregate_matrix

pegasus aggregate_matrix allows aggregating arbitrary matrices with the help of a CSV format sample sheet.

Type:

```
pegasusio aggregate_matrix -h
```

to see the usage information:

```
Usage:
  pegasusio aggregate_matrix <csv_file> <output_name> [--restriction <restriction>...␣
↪options]
  pegasusio aggregate_matrix -h
```

- Arguments:

  csv_file
  Input csv-formatted file containing information of each sc/snRNA-seq sample. This file must contain at least 2 columns - Sample, sample name and Location, location of the sample count matrix in either 10x v2/v3, DGE, mtx, csv, tsv or loom format. Additionally, an optional Reference column can be used to select samples generated from a same reference (e.g. mm10). If the count matrix is in either DGE, mtx, csv, tsv, or loom format, the value in this column will be used as the reference since the count matrix file does not contain reference name information. In addition, the Reference column can be used to aggregate count matrices generated from different genome versions or gene annotations together under a unified reference. For example, if we have one matrix generated from mm9 and the other one generated from mm10, we can write mm9_10 for these two matrices in their Reference column. Pegasus will change their references to 'mm9_10' and use the union of gene symbols from the two matrices as the gene symbols of the aggregated matrix. For HDF5 files (e.g. 10x v2/v3), the reference name contained in the file does not need to match the value in this column. In fact, we use this column to rename references in HDF5 files. For example, if we have two HDF files, one generated from mm9 and the other generated from mm10. We can set these two files' Reference column value to 'mm9_10', which will rename their reference names into mm9_10 and the aggregated matrix will contain all genes from either mm9 or mm10. This renaming feature does not work if one HDF5 file contain multiple references (e.g. mm10 and GRCh38). csv_file can optionally contain two columns -

nUMI and nGene. These two columns define minimum number of UMIs and genes for cell selection for each sample. The values in these two columns overwrite the min_genes and min_umis arguments. See below for an example csv:

```
Sample,Source,Platform,Donor,Reference,Location
sample_1,bone_marrow,NextSeq,1,GRCh38,/my_dir/sample_1/filtered_gene_bc_
↪matrices_h5.h5
sample_2,bone_marrow,NextSeq,2,GRCh38,/my_dir/sample_2/filtered_gene_bc_
↪matrices_h5.h5
sample_3,pbmc,NextSeq,1,GRCh38,/my_dir/sample_3/filtered_gene_bc_
↪matrices_h5.h5
sample_4,pbmc,NextSeq,2,GRCh38,/my_dir/sample_4/filtered_gene_bc_
↪matrices_h5.h5
```

output_name
The output file name.

- Options:

-\-restriction <restriction>…
Select data that satisfy all restrictions. Each restriction takes the format of name:value,…,value or name:~value,..,value, where ~ refers to not. You can specifiy multiple restrictions by setting this option multiple times.

-\-attributes <attributes>
Specify a comma-separated list of outputted attributes. These attributes should be column names in the csv file.

-\-default-reference <reference>
If sample count matrix is in either DGE, mtx, csv, tsv or loom format and there is no Reference column in the csv_file, use <reference> as the reference.

-\-select-only-singlets
If we have demultiplexed data, turning on this option will make pegasusio only include barcodes that are predicted as singlets.

-\-min-genes <number>
Only keep cells with at least <number> of genes.

-\-max-genes <number>
Only keep cells with less than <number> of genes.

-\-min-umis <number>
Only keep cells with at least <number> of UMIs.

-\-max-umis <number>
Only keep cells with less than <number> of UMIs.

-\-mito-prefix <prefix>
Prefix for mitochondrial genes. If multiple prefixes are provided, separate them by comma (e.g. “MT-,mt-“).

-\-percent-mito <percent>
Only keep cells with mitochondrial percent less than <percent>%. Only when both mito_prefix and percent_mito set, the mitochondrial filter will be triggered.

-\-no-append-sample-name
Turn this option on if you do not want to append sample name in front of each sample’s barcode (concatenated using ‘-‘).

-h, -\-help
Print out help information.

- Outputs:

output_name.zarr.zip
A zipped Zarr file containing aggregated data.

- Examples:

```
pegasusio aggregate_matrix --restriction Source:BM,CB --restriction Individual:1-8 -
↪-attributes Source,Platform count_matrix.csv aggr_data
```

24.4 Convert to/from `AnnData`

PegasusIO stores data as *MultimodalData* objects, and each *MultimodalData* has a default *UnimodalData* object to refer to. Each time, a format conversion is to convert the default *UnimodalData* to another format.

AnnData is the annotated data matrix object provided by anndata package.

Let `mmdata` be a PegasusIO *MultimodalData* object. Use the following code to convert its default *UnimodalData* to *AnnData*:

```
>>> adata = mmdata.to_anndata()
```

And `adata` is the wanted *AnnData* object.

Now let `adata` be an *AnnData* object (See anndata reading documentation for how to load file to *AnnData*). Use the following code to convert it to PegasusIO *MultimodalData*:

```
>>> import pegasusio as io
>>> mmdata = io.MultimodalData(adata)
```

And `mmdata` is the wanted *MultimodalData* object.

24.5 Tutorials

PegasusIO Tutorial